Problem (1)

Given a binary pattern in some memory location, is it possible to tell whether this pattern represents a machine instruction or a number?

Problem (2)

Write a program that can evaluate the expression

$$A \times B + C \times D$$

in a single-accumulator processor. Assume that the processor has Load, Store, Multiply, and Add instructions, and that all values fit in the accumulator.

Problem (3)

Byte-Sorting Program (Page 87)

Sorting a list of n bytes stored in memory into ascending alphabetical order using the Selection Sort Algorithm.

- C-Language Program

```
for (j=n-1; j > 0; j=j-1)
{
        for (k=j-1; k >= 0; k=k-1)
        {
                if (List[k] > List[j])
                {
                        Temp = List[k];
                        List[k] = List[j];
                        List[j] = Temp;
                }
        }
}
```

Problem (4)

Both of the following statements cause the value 300 to be stored in location 1000, but at different times.

```
ORIGIN      1000
DATAWORD    300
```

and

```
Move    #300,1000
```

Explain the difference.

Problem (5)

Rewrite the assembly program to compute the dot product of two vectors A, B of n-bits using a subroutine.

Problem (6)

Let the address stored in the program counter be designated by the symbol X1. The instruction stored in X1 has an address part (operand reference) X2. The operand needed to execute the instruction is stored in the memory word with address X3. An index register contains the value X4. What is the relationship between these various quantities if the addressing mode of the instruction is

a) Direct. b) indirect. c) PC relative. d) indexed

Solution:

a) X3=X2

b) X3=(X2)

c) X3=X1+X2+1

d) X3=X2+X4

Problem (7)

11.3 An address field in an instruction contains decimal value 14. Where is the corresponding operand located for:

a) immediate addressing?

b) direct addressing?

c) indirect addressing?

d) register addressing?

e) register indirect addressing?

Problem (8) (Problem 2.18)

A FIFO queue of bytes is to be implemented in the memory, occupying a fixed region of k bytes. You need two pointers, an IN pointer and an OUT pointer. The IN pointer keeps track of the location where the next byte is to be appended to the queue, and the OUT pointer keeps track of the location containing the next byte to be removed from the queue.

(*a*) As data items are added to the queue, they are added at successively higher addresses until the end of the memory region is reached. What happens next, when a new item is to be added to the queue?

(*b*) Choose a suitable definition for the IN and OUT pointers, indicating what they point to in the data structure. Use a simple diagram to illustrate your answer.

(*c*) Show that if the state of the queue is described only by the two pointers, the situations when the queue is completely full and completely empty are indistinguishable.

(*d*) What condition would you add to solve the problem in part *c*?

(*e*) Propose a procedure for manipulating the two pointers IN and OUT to append and remove items from the queue.

Problem(9) (Problem 2.19)

Consider the queue structure described in Problem 2.18. Write APPEND and REMOVE routines that transfer data between a processor register and the queue. Be careful to inspect and update the state of the queue and the pointers each time an operation is attempted and performed.